L Number	Hits	Search Text	DB	Time stamp
-	269	monitor$3 same (fib$3 adj2 channel$1)	USPAT; US-PGPUB	2002/12/03 13:47
-	156	(monitor$3 same (fib$3 adj2 channel$1)) and @ad<19991210	USPAT; US-PGPUB	2002/12/03 14:47
-	96	((monitor$3 same (fib$3 adj2 channel$1)) and @ad<19991210) and (id$1 or identif$11)	USPAT; US-PGPUB	2002/12/03 13:49
-	493	host adj2 bus adj2 adapter$2	USPAT; US-PGPUB	2002/12/03 14:46
-	450	(host adj2 bus adj2 adapter$2) and controller$1	USPAT; US-PGPUB	2002/12/03 14:46
-	73	((host adj2 bus adj2 adapter$2) and controller$1) and (fib$3 adj2 channel$1)	USPAT; US-PGPUB	2002/12/03 14:47
-	30	(((host adj2 bus adj2 adapter$2) and controller$1) and (fib$3 adj2 channel$1)) and @ad<19991210	USPAT; US-PGPUB	2002/12/03 14:47

US-PAT-NO: 6343324

DOCUMENT-IDENTIFIER: US 6343324 B1

TITLE: Method and system for controlling access share storage devices in a network environment by configuring host-to-volume mapping data structures in the **controller** memory for granting and denying access to the devices

---------- KWIC ---------

Method and system for controlling access share storage devices in a network environment by configuring host-to-volume mapping data structures in the **controller** memory for granting and denying access to the devices

The invention provides structure and method for controlling access to a shared storage device, such as a disk drive storage array, in computer systems and networks having a plurality of host computers. A method for controlling access to a hardware device in a computer system having a plurality of computers and at least one hardware device connected to the plurality of computers. The method includes the steps of associating a locally unique identifier with each the plurality of computers, defining a data structure in a memory identifying which particular ones of the computers based on the locally unique identifier may be granted access to the device; and querying the data structure to determine if a requesting one of the computers should be granted access to the hardware device. In one embodiment, the procedure for defining the data structure in memory includes defining a host computer ID map data structure in the memory; defining a port mapping table data structure comprising a plurality of port mapping table entries in the memory; defining a host identifier list data structure in the memory; defining a volume permission table data structure in the memory; and defining a volume number table data structure in the memory. In one particular embodiment, the memory is a memory of a memory **controller** controlling the hardware device, and the hardware device is a logical volume of a storage subsystem. The invention also provides an inventive **controller** structure, and a computer program product implementing the inventive method.

Conventional operating systems may typically assume that any storage volume or device is "private" and not shared among different host computers. In a distributed computing system, such as a network server system, a disk drive, a storage volume, a logical volume, or other storage device may be shared and represent common storage. When a **controller** responsible for controlling read, write, or other access to the storage device, such as a hard disk array **controller** (for example a RAID **controller**) is attached to the plurality of host computers, such as through a SCSI Bus, **Fibre Channel** Loop, or other storage device interface, problems may arise because one or more of these plurality of host computers may overwrite or otherwise corrupt information needed for the correct operation of another different host computer system.

With respect to FIG. 1, we now describe an exemplary distributed computing system 100 having first, second, and third host computers 101 (101-1, 101-2, 101-3) coupled to an array **controller** 104 which in turn is coupled to a storage subsystem 108 formed from one or more logical volumes, here shown as an array of logical disk drive storage volumes (108-1, 108-2, 108-3, . . . , 108-N). In general, these logical volumes 108 may correspond to physical hard disk drive devices, or to groups of such physical hard disk drive devices. In this embodiment, the three host computers 101-1, 101-2, and 101-3 are coupled to array **controller** 104 via a **Fibre Channel** Loop 120 communications channel, and

the logical volumes 108 of the storage subsystem are coupled to the array **controller** 104 via an appropriate channel 122, such as for example either a **Fibre Channel** Loop communications channel or a parallel SCSI communications channel. For the **Fibre Channel** Loop, SCSI protocols are frequently used in addition to the **Fibre Channel** physical layer and related protocols and standards. **Fibre Channel** Loop 120 is advantageous for interconnections of the host computers because of the flexibility and extensibility of this type interface to a large number of host computers and also, with respect to the inventive structure and method, for the existing support of World Wide Number (WWN) identification.

In computing system 100, array **controller** 104 divides the storage into a number of logical volumes 108. These volumes are accessed through a Logical Unit Number (LUN) addressing scheme as is common in SCSI protocol based storage systems, including SCSI protocol based **Fibre Channel** Loop physical layer configurations. The term LUN refers to a logical unit or logical volume, or in the context of a SCSI protocol based device or system, to a SCSI logical unit or SCSI logical volume. Those workers having ordinary skill in the art will appreciate that the number of physical disk drives may be the same as, or different from, the number of logical drives or logical volumes; however, for the sake of simplicity and clarity of description here we use these terms interchangeably, focusing primarily on logical volumes as compared to physical disk drives. The manner in which physical devices are generically assigned, grouped, or mapped to logical volumes is known in the art and not described further here.

Therefore there exists a need for structure and method that resolves this shared access problem by efficiently testing and validating authorization to access a storage volume, logical volume, or storage device on the array **controller** to a specific set of host computers and limiting access only to authorized hosts, so that neither critical information nor data generally will be overwritten or otherwise corrupted.

In one embodiment, the procedure for defining the data structure in memory includes defining a host computer ID map data structure in the memory; defining a port mapping table data structure comprising a plurality of port mapping table entries in the memory; defining a host identifier list data structure in the memory; defining a volume permission table data structure in the memory; and defining a volume number table data structure in the memory. In one particular embodiment, the memory is a memory of a memory **controller** controlling the hardware device, and the hardware device is a logical volume of a storage subsystem.

The invention also provides an inventive **controller** structure, and a computer program product implementing the inventive method.

FIG. 1 is an illustration showing an embodiment of a distributed computing system having a plurality of host computers coupled to an array of logical storage volumes though a **Fiber Channel** Loop and an a disk array **controller**.

The invention includes method, apparatus, system, and computer program product for providing controlled access to storage volume(s) on an inventive storage system **controller,** such as a hard disk drive array **controller** 106. The inventive structure, method, and computer program product, including **controller** 106 and storage subsystem 108 having access controls, further solves the access and security problem of conventional systems and methods by limiting access to a volume of storage 108 on, or controlled by, the array **controller** 106 to a specific set of host computers 101, as identified by a unique identifier (for example the World Wide Name (WWN) 107 associated with the host computer 101 via

its network interface or by using other identifying means, so long as that identifying means is unique among the interconnected devices. The invention also provides limited security control of data, where access to data must be limited or shielded from other users of the system. Further, the inventive method accomplishes this task with a minimal number of searches and overhead, and with minimal performance degradation. Where a particular host has two or more interfaces to the **controller,** including for example a host having multiple **host bus adapters** each having a unique ID, access may advantageously be further controlled based on the interface ID instead of, or in addition to, the host ID.

The inventive structure and method are particularly suitable for situations where one or more data storage array **controllers** are attached to multiple host computers (or to a single host having multiple interfaces) with a **controller** that will request data transactions with the storage devices, such as for example a Redundant Array of Independent Disk (RAID) storage device array. As already described, the problem exists for both homogeneous and heterogeneous combinations of host computer hardware and host computer operating systems, where here heterogeneity refers to either differences in hardware type or operating system. Usually heterogeneity of the hardware is not a significant issue, as this is resolved through the standards that define SCSI and **Fibre channel** operation. Heterogeneity of the operating system may more likely result in corrupted data.

In one aspect the inventive structure Host-to-Volume Mapping (HVM) feature restricts access to any particular configured Logical Volumes only to a single host or group of hosts. This provides access and security control of data for the storage array, and is particularly advantageous for maintaining data integrity in a Storage Area Network (SAN) environment, where multiple hosts are connected to a **controller,** frequently to an external **controller**. Storage Area Network (SAN) refers to a collection of one or more host computers attached to a common pool of storage resources. The Host-to-Volume Mapping (HVM) feature is desirably implemented as a software and/or firmware computer program product executing in a processor or CPU within the **controller** 106 and utilize data structures defined in a memory associated with the processor to alter the operation of the **controller**. In general **controller** 106 of the invention may differ from conventional **controller** 104; however, the invention may also be used with conventional **controller** suitably modified to provide the characteristics described herein. For example, the inventive computer program product may be stored and executed in **controllers having a fibre channel** host interface and appropriate memory for defining and storing the inventive data structures, such as for example, the Mylex Corporation DACSF, DACFF, and DACFFX **controllers,** as well as other **controllers**.

Aspects of **controller** design for certain exemplary Mylex Corporation **controllers** are described in the DACSX / DACSF / DACFL / DACFF OEM System Reference Manual Firmware Version 5.0--Mylex Corporation Part Number 771992-04 (Mylex Corporation of Fremont, Calif. and Boulder, Colo.), and herein incorporated by reference. SCSI-3 Primary Commands, such as commands that are applicable to **fibre channel** connected devices using SCSI-3 commands are described in SCSI-3 Primary Commands, T10/995D revision 11a--Mar. 28, 1997, also hereby incorporated by reference.

By utilizing the inventive HVM, each Logical Volume 108 may be configured to be visible to a single one of host computers 101, (for example to host 101-2 only) or to a selected group or set of host computers (for example to hosts 101-1 and 101-2 only.) Referring to the hardware configuration of FIG. 1, one simple HVM configuration would allow host computer 101-1 access to Logical Volume 108-1 only, host computer 101-2 access to logical volume 108-2 only, and host

computer 101-3 access to logical volume 108-3 only; even though all hosts 101 and all logical volumes 108 are physically connected to the **controller**.

As described in greater detail herein, **controller** 106 uses novel data structures and a node name, such as the World Wide Name (WWN), associated with each **fibre channel** loop 120 device, including a **Fibre Channel Host Bus Adapter** installed in each host computer 101, to uniquely identify the host computers that have logged into **controller** 106. (A list of valid host computers that have been granted access to each logical volume, and their corresponding WWNs, may optionally be provided to external configurators, to provide a graphical user interface to assist in configuring the **controller** 106 and to configure the HVM.) Note that a node is one or a collection of more than one port and that a Node Name generally refers to a Wold Wide Name (WWN) identifier associated with a node. Port Name is a World Wide Name identifier associated with a port, for example a port at which a host or a logical volume couples to **controller** 106.

The inventive HVM structure, method, and computer program product provides a solution to the afore described shared access problem by utilizing a unique host identifier (host node name identifier) in conjunction with other structures and procedures to control access to storage on each logical volume. As the use of a host node name identifier is important to the operation of the invention, and as World Wide Names (WWNs) are an existing useful type of host node name identifier particularly for **Fibre Channel** Loop connected hosts and **controllers,** we briefly describe some attributes of WWN before proceeding with a more detailed description of HVM.

When the communications channel 120 coupling hosts 101 to array **controller** 106 is a **Fiber Channel** Loop compliant with the "**Fibre Channel** Physical and Signaling Interface" ((FC-PH) Rev. 4.3, X3T11, Jun. 1, 1994 standard, American National Standards for Information Systems), which standard is hereby incorporated by reference, each device on the loop 120 including each host 101 by virtue of a **Fiber Channel Host Bus Adapter** has a unique identifier, referred to as its World Wide Name (WWN) 107. WWN 107 are known in the art, particularly for **Fibre Channel** devices, and we only describe in detail aspects of the WWN that are useful in understanding the structure and operation of the invention.

A World Wide Name (WWN 107) is a 64-bit identifier (8-byte), with a 60-bit value preceded by a 4-bit Network Address Authority Identifier (NAAI), used to uniquely identify devices, nodes, or ports, including for example a **Host Bus Adapter** (HBA), for connecting a host computer 101 to a **Fibre Channel** communications loop 120. This WWN 107 is unique for each **fibre channel** device, usually in the form of a number (serial number) that the manufacturer registers with the appropriate standards committee through the process defined as a part of the **Fibre Channel** standards specification. It is unique to each **fibre channel** connect device manufactured. For example the **fiber channel** interface card or host bus adaptor (HBA) in each host has a unique WWN. While there are many fields and subfields in this character or number WWN string, from the standpoint of the invention, many of the fields and subfields are irrelevant, and for the purposes of the invention the WWN is conveniently thought of as a unique serial number for the **fibre channel** device. Detail of the format and content of the WWN are described in the "**Fibre Channel** Physical and Signaling Interface (FC-PH) Rev. 4.3, X3T 11, Jun. 1, 1994 standard, American National Standards for Information Systems (ANSI)", hereby incorporated by reference.

The WWN 107 is used to uniquely identify each host computer 101 connected to the **Fibre Channel** loop 120, or more specifically each **Host Bus Adapter** (HBA) coupling the **fibre channel** bus 120 to the processor and memory system in the host computer. Thus, if there are two **fibre channel** HBAs installed in a single

host computer 101, that host computer will have two WWNs associated with it, and it will be possible to identify not only which host, but also which HBA of the host the communication was sent from or should be directed to in a response. As the WWNs are universal and currently exist, an aspect of the invention lies in the use of WWNs to allow access to a volume of storage based on the WWN 107. Furthermore, at least some embodiments of the invention may be implemented in existing hardware, while other embodiments benefit from or require specific **controller** hardware not provided in conventional **controllers**.

It is noted that while we describe the invention primarily relative to **Fibre channel** loops 120 and the WWN 107 associated with such **Fibre channel** loops, the invention is not limited to such **Fibre-channel** loops or to WWNs as the only host node identifier, and can be used with alternative communication channel strategies and protocols and/or with different host node identifiers, such as for example for parallel SCSI channels and SCSI IDs, although this would not represent a preferred configuration due to the limited number of SCSI addresses (15) and the limited physical distance (usually about 6 meters) between the SCSI devices, neither of which limitations are present in a **Fibre Channel** implementation. For example, various computing node identifiers may be envisioned for computers and storage volumes interconnected over the Internet or world wide web.

With reference to FIG. 2, we now describe an embodiment of the inventive Host Volume Mapping (HVM) structure and method in a system configuration in which a plurality of host computers 101-1, 101-2, . . . , 101-M attach to one or more external storage device array **controllers,** hereinafter "**controller**" 106, and a plurality of magnetic hard disk drives configured as a plurality of logical volumes 108 coupled to the **controller**; a configuration frequently used to implement a Storage Area Network (SAN) configuration. Host computers 101 are attached to **controller** 106 through a **Fibre Channel** arbitrated loop 120 and/or through a switch (of which many types are known). Logical Volumes 108 may be coupled to **controller** 106 using either **Fibre Channel** Arbitrated Loop 122, or where sufficient to support the number of units and the cable length limitations, via parallel SCSI chain. Logical volumes 108 may be configured as RAID or other storage subsystems as are known in the art.

We now describe an embodiment of the inventive Access Control and Validation Procedure (ACVP) 300 with reference to the computer system 201 in FIG. 2. FIG. 2a shows **controller** 106 and its relationship to host computers 101 and logical volumes 108, and further including a top-level illustration of the data structures defined in NVRAM 182 of the **controller**. FIG. 2b shows additional detail of the data structure. The phrase "data transaction" as used here refers to information transfers between a host 101 and the array **controller** 106 and includes such typical operations as reading and writing data to the array **controller** 106. A data transaction starts or is initiated by a host computer 101 when it issues a data transfer command (typically read or write request) over the **fibre channel** bus 120.

Each logical volume 108 in the storage array is assigned or associated with a volume data structure 140, one element of which is a Volume WWN Table (VNT) 130. These VNT tables (130-1, 130-2, . . . , 130-N) may be thought of as separate small tables or as a single larger table, but in any event provide a VNT data structure used in subsequent search or query operations. (We will later expand the description of the concept of the Volume data Structure to encompass a Volume Permission Table 160.) For example, Logical Volume 108-1 is associated with VNT 130-1, Logical Volume 108-2 is associated with VNT 130-2 and Logical volume 108-N is associated with VNT 130-N. This or these VNT tables are stored as a part of **controller** 106 configuration data stored in a non-volatile memory (NVRAM) 182 of **controller** 106 and desirably on the disks

(logical volumes) associated with that **controller** 106. This configuration, typically referred to as a "Configuration on Disk" (COD) can be accessed (written and read) by vendor unique direct commands which permit the storage volume array 108 to be initially configured and/or reconfigured as necessary. These Vendor Unique commands are described elsewhere in this specification.

Each Volume WWN Table 130 has a finite number of entries at any given time, one entry for each WWN that is permitted to access its associated logical volume 108. But, while the size or number of entries in any one Volume WWN Table 130 is finite at any given time, the finite number corresponding to the number of **fibre channel** devices (hosts or HBAS) that are permitted to access the volume, the size of the Volume WWN Table is not fixed and can be expanded when necessary to any size so as to accommodate the required number of **fibre channel** device entries, limited only in a practical sense by the memory available to store the entries. In one embodiment, if all of the entries in the VNT associated with the logical volume are zero, the zero value serves as an indication that all hosts may have access to that particular logical volume.

At the start of a data transaction, a host computer 101 desiring to access a logical volume 108 controlled by **controller** 106 must login or otherwise identify its access request. Host 101 first logs in to the logical volume storage array 108 via **controller** 106, then makes requests to access a specific logical volume. Aspects of this login are a conventional part of the **fibre channel** arbitrated loop protocol and not described here in detail. (See **Fibre Channel** Arbitrated Loop Protocol Standard and **Fibre Channel** Physical and Signaling Interface, which are herein incorporated by reference.) As a part of this login transaction, the array **controller** 106 is notified that a host 101 is attempting to connect to the logical volume(s) 108 and the unique WWN 107 and Loop ID 152 corresponding to the requesting host 101 (or HBA 103 associated with the host 101) is communicated in the form of a command packet 109.

As the login procedure continues, **controller** 106 identifies a Host Index (HI) 151 for that host based on the received Loop ID 152. In one embodiment, the Host Index 151 is generated by the **controller** sequentially based on the order of the hosts attempt to login to a **Fibre Channel** port. The first host to attempt a login will be assigned HI=0, the second host will receive HI=1, and so on. Other host computer to HI assignment rules may alternatively be implemented. The Host Index 151 functions or operates as a pointer to allow simplified access to information stored in the Host WWN List 153 as well as indirectly into the Volume WWN Tables (VNT) 130 and Volume Permission Tables (VPT) 194 as described in greater detail below. In one embodiment, the Host Index consists of 4 bits, so that at least 16 different hosts can be uniquely identified, while other embodiments provide a larger number of bits and permit a greater number of hosts to be uniquely identified.

We note that prior to this attempted login, a first list of WWNs 107 of host computers 101 that have previously logged in to **controller** 106 is stored in a Host WWN List 153 data structure defined in memory (NVRAM) 182 of **controller** 106 and indexed by a Host Index 151. For example, in FIG. 2, Host WWN List 153 includes indexed storage for up to 256 (numbered 0-255) host WWN 107 entries in a linear list. The WWN entries in the list (for example, the entry corresponding to HI=0 showing "20.00.00.E0.8B.00.00.07" hexadecimal) are exemplary and do not necessarily bear any relationship with past, present, or future actual WWN associated with manufactured devices. Storage locations in Host WWN List 153 that are empty are indicated by "FF.FF.FF.FF.FF.FF.FF.FF".

A second or Host ID Map List 155 data structure storing a list of Host Indices 151 also defined in memory (NVRAM) 182 of **controller** 106 is indexed by **Fiber Channel** Loop ID 152. This Host ID Map List 155 maps each Loop ID 152 to a Host

Index 151 as illustrated in FIG. 2b. In one embodiment, the Loop IDs 152 in consecutive memory storage locations are consecutive numbers (the pointer), while the Host Index 151 values are not consecutive and are represented by two-byte hexadecimal values.

In the embodiment of FIG. 2, **controller** 106 maintains only one Host WWN List 153 for all host ports 114, 184; and even for the case of multiple **controllers** 106, this structure and procedure allows for the simplest representation of the **fibre channel** connection topology in that a particular host computer's Host Index (HI) 151 remains the same regardless of the port or **controller** the host is communicating with.

Once the Host WWN List 153 and Host ID Map list 155 have been established as the host computers login to the array **controller** 106, and the Volume WWN Table 130 is generated by the array **controller** as a result of these logins, the procedure is able to validate or alternatively deny access to a host attempting the login.

We now describe an exemplary LUN-to-Logical Volume Mapping (Volume Mapping) Structure and procedure. "Volume Mapping" (VM) is a process where a **controller** 106 maps a particular **Fibre Channel** (as identified by the I/O processor on which the command is received), **Fibre Channel** Loop ID, and SCSI LUN to a particular Logical Volume. A SCSI LUN is a path to a logical volume of storage. "Host-to-Volume Mapping (HVM)" extends the concept, method, and structure of Volume Mapping (VM) by allowing a particular Host (identified by the **Fibre Channel, Fibre Channel** Loop ID, and SCSI LUN) to a logical volume. HVM therefore permits host access control to volumes while VM does not. According to **fibre channel** conventions, the host computer loop ID is assigned based on hardware, software, or negotiated settings, but other assignment rules may alternatively be used in conjunction with the invention structure and method.

The Volume Mapping feature maintains a Volume Mapping Table in the form of a Port Mapping Table 190 for **controller** 106, port 114, **Fibre Channel** I/O Processor 184, and Logical Volume 108 combinations. This allows a specific Logical Volume to appear as a different LUN on each host port 114. There may be a plurality of host ports 114 and **Fibre Channel** I/O Processors 184 for each port, and each host port is associated with the particular **controller** 106. There may generally be multiple host computers attached to the **controller** host port or ports by virtue of the characteristics of the **Fibre Channel** loop or the parallel SCSI protocols and/or specifications. Allowing a specific logical volume to appear as a different LUN on each host port is advantageous because it permits great flexibility in allowing host access to the logical volume, and for a HVM environment described in greater detail hereinafter, this feature is particularly advantageous because the permitted flexibility allows storage volume mapping to a heterogeneous collection of host computers with heterogeneous operating systems. Each of these systems will have specific requirements for mapping storage and accommodating these different storage mapping requirements advantageously relies on the ability to map storage in a variety of different ways.

The idea of Volume Mapping is to break up the storage capacity of the physical disc drives connected to the array **controller** into "Logical Volumes", then to control the host computer's access to these logical volumes by assigning an access path to each logical volume and checking to verify that an attempted access path is valid. Typically, the access path consists of a host-to-**controller** Port 114 (i.e., which Host I/O Processor 184), the SCSI ID (or fibre loop ID) of the Host Processor 184, and the SCSI LUN Number of the read or write command.

The Port Mapping Table Entries 191 contained within the Port Mapping Table 190 are advantageously instantiated for each **controller** 106, each host channel 184, and each logical volume 108 as illustrated in FIG. 2b, and defines how each host port 114 and host port I/O **controller** 184 connects through array **controller** 106 to each logical volume 108. The Port Mapping Table 190 contains a plurality of Port Mapping Table entries 191, one entry for each **controller** 106, Host I/O processor 184, and Logical Volume 108 combination. Each Port Mapping Table entry 191 includes an 8-bit (1-byte) Target ID 192 containing the loop ID of the Logical Volume on this port, an 8-bit (1-byte) LUN 193 containing the LUN number for the logical volume on this port to which the command is directed (also referred to as the target loop ID), a 32-byte Volume Permission Table 194, and a Flag Bit 195 field (8-bits) storing various flag indicators.

The Port Mapping Table Entries 191 are advantageously instantiated for the **controller** 106 in a single **controller, or for each controller** in the case of a multi-**controller** (e.g. duplex-**controller**) configuration, for each host channel, and for each logical volume. This means that there is a Port Mapping Table 190 that defines how each host port (the Host Computer **Fiber Channel** I/O Processors 184 in **controller** 106) connects to each logical volume.

Once the Logical Volumes 108 are configured, the **controller** 106 maintains a Volume Permission Table 194 in its processor memory 182 for each Logical Volume containing a list of WWNs for hosts permitted access to the logical volume. This table identifies which of the host computers 101 are granted access to each particular Logical Volume 108 coupled to the **controller** based on the WWNs. A **controller** 106 may typically have a plurality of host ports 114 and disk drive ports 115 and associated I/O processors 184, 185 at each port. The I/O processors such as host I/O processors, **Fibre Channel** I/O processors 184, 185 can be the same type, but these are segregated into host ports 184 (for communication with the host computer) or disk ports 185 (for communication with the disks.)

In one embodiment of the invention, an Intelligent SCSI Processor (ISP) chip is used for **Fibre Channel** I/O Processors 184, 185. The ISP processor chip is manufactured by Q-Logic Corporation and is available from Q-Logic Corporation of 3545 Harbor Blvd, Costa Mesa, Calif. 92626. Several variations of ISP chips are manufactured by Q-Logic in the "ISP product family".

Controller 106 uses the LUN number requested by the host and the identity of the host-to-controll**er port 114**, 115 (or 184, 185) at which the command was received, both of which are produced by the ISP with command, to determine which Logical Volume the host is trying to access. The operation of Fibre Cha**nnel protocol** chips, such as ISP, is known in the art and not described in further detail here.

Controller 106 uses the Volume WWN Table 130 to determine allowed and disallowed access to a specific logical volume by any particular host computer. If a host computer 101 sends a new command to controlle**r 106, the** controlle**r validate**s the WWN, controlle**r port, an**d LUN against data in the table 130 prior to servicing the host command. If the WWN, LUN, and host-to-controll**er port in**formation are valid for the Logical Volume, the command requested by the host is completed normally. However, if the WWN, LUN, and host-to-controll**er port com**bination are not valid for the logical volume, the requested command is not completed normally and a status is returned indicating that the particular logical volume is not supported. (However, three exceptions occur and are described below.) In one particular embodiment of the invention, program code implemented as firmware 301 provides that a host command that cannot be

validated is completed with a "Check Condition" status, with the sense key set to "Illegal Request (05h)" and the sense code set to "Logical Unit Not Supported (25h)".

We now focus this description toward an embodiment of the inventive Host Volume Mapping (HVM) structure and method. In this context a diagram of the various data structures, lists, bit maps, and the like present on the **controller** 106, host 101, and logical volumes 108 along with their relationships to each other are illustrated in FIG. 2b. The existence of the WWN 107 for each **Fiber Channel** Loop 120 coupled host computer 101 provides an opportunity to utilize the WWN in the inventive method to establish a separate table of allowed WWNs, the Volume WWN Table 130 for each logical volume 108. Access to each particular logical volume 108 is permitted by disk array **controller** 106 only when the WWN of the particular host computer requesting data from the particular logical volume 108 is contained in the Volume WWN Table 130 associated with the particular logical volume 108. The WWN must be present in the table, and if it is present and the host has logged onto the array, only a check of the Volume Permission Table is further required to validate access. For an array of N logical volumes, N Volume WWN Tables 130 (130-1, 130-2, 130-3, . . . , 130-N) are provided in the system. If all volumes may be accessed by the identical set of host computers, each of the N Volume WWN Tables will contain the same list of host WWNs; however, the contents of the N Volume WWN Tables 130 for the logical volumes will generally differ when different volumes are available for access by different hosts.

At this point array **controller** 106 searches all of the Volume WWN Tables 130 associated with each logical volume (that is Tables 130-1, 130-2, . . . , 130-N) to determine which, if any, of the logical volumes the requesting host has permission to access. A host will have permission to access a logical volume when that host's world wide name appears in the Volume WWN Table 130 associated with that logical volume and will not have permission to access a logical volume when that host's world wide name does not appear in the table. Thus the array **controller** controls access.

When the host computer 101 attempts to read or write a logical volume 108, the HI 151 for the requesting host is determined by **controller** 106 based on that hosts **Fibre channel** Loop ID 152 which is returned by the **Fibre channel** I/O processor 184 along with detailed information that fully defines the operation, including the LUN to which the read or write request is addressed. If the request is not a Vendor Unique command (which might indicate an attempt to configure or reconfigure a volume and require special handling), the array **controller** 106 examines the Volume Permission Table 194 for that HI and for that logical volume. If the permission indicator associated with that HI is true ("1"), the request is executed normally. That is, the read, write, or other access request is executed using the normal procedure for reading or writing data to or from the logical volume. On the other hand, if the permission indicator associated with that HI and for the logical volume to which the request is addressed is false ("0"), then the read or write command is rejected back to the host computer from which it was sent with an error condition.

Special conditions exist when the request is either an "Inquiry" command, a "Vendor Unique" command, or a "Report LUNs" command. These commands are generally associated with determining the configuration or characteristic of the system, or with configuring or reconfiguring the system or components thereof, such as the **controller** 106. We describe aspects of these special commands in greater detail elsewhere in this specification. For other than Inquiry, Vendor Unique, and Report LUNs type commands, if a request is made by a host for a logical volume and the logical volume does not have permission for

that host, the array **controller** will assert an error condition and deny access.

If the host has permission and the command is neither an Inquiry, nor a Vendor Unique, nor a Report LUNs command, the command is processed normally. Normal processing of a read command means that upon receipt of a read command the array **controller** will read the data from the attached disk drive or drives (logical volumes) and return this data to the host. Upon receipt of a write command the array **controller** will store the data sent by the host to the attached disk drive or drives.

In addition to these procedural steps the **controller** 106 should also verify that the logical volume is mapped to the **controller** port on which the command was received. As there can be multiple host-to-**controller** ports 114, 184; and a logical volume can be mapped to any single one of the ports, to any selected plurality of the port, or to none of the ports; the **controller** 106 should assure that the logical volume can be accessed through the particular host-to-**controller** port on which the command was received. The **controller** should also verify that the logical volume is mapped to the Logical Unit Number (LUN) of the command. Since each port can have many logical units as defined in the SCSI and **Fibre Channel** specifications, this allows one port to access many devices. Finally, the **controller** should verify that the WWN is valid for this logical volume, as already described.

Frequently, the configuration of the array **controller**(s) 106 is stored on a special reserved area on the disks, this is referred to as "Configuration on Disk" (COD). This permits more efficient array **controller** 106 configuration when an array **controller** is replaced (such as for example after a **controller** failure). The replacement **controller** can retrieve the original configuration from the disk and automatically restore it rather than having to figure out its configuration information during a separate and time consuming reconfiguration procedure. Where Configuration on Disk (COD) space is limited, the maximum number of connected hosts may be limited, for example, to some number of hosts, such as to sixteen hosts. In other embodiments where COD is not limited, the maximum number of connected hosts parameter may be set to 256 entries so as to allow a sufficient number of entries for a fully populated loop in accordance with the **fibre channel** specification.

In addition to the Volume WWN Table 130, firmware in array **controller** 106 uses the Host ID Map 155 to translate from a host computer's **fibre channel** loop ID 152 to the correct Volume WWN Table 130 entry. This allows hosts 101 to change their particular **fibre channel** loop ID 152 without affecting the Volume WWN Table 130. A Host ID Map 155 is maintained for each **fibre channel** port on array **controller** 106. The maximum number of **fibre channel** host node (WWN) names that can be accommodated is set to 256 to allow any loop ID in the range of 0 to 255.

The first time a **controller** 106 is booted with firmware containing the HVM feature, following the first Loop Initialization Primitive (LIP) which resets the **Fibre Channel,** the firmware executing in the **controller** 106 retrieves the login information from all hosts 101 on the loop 120. From the login information, the firmware constructs the Volume WWN Table 130 as well as the Host ID Map Table 155. These two tables in tandem provide the firmware the capability to correctly translate the loop ID 152 embedded in a new command from the fibre protocol chip (e.g. ISP chip) to the Host Index 151, which in turn identifies a host 101 with a specific WWN 107. Effectively, the loop ID 152 is mapped to the host WWN 107 by: (i) first mapping the loop ID 152 to the Host Index 151, and (ii) then mapping the Host Index 151 to the host WWN 107. This approach is advantageous because only a small (minimum) number of searches and comparisons are needed to determine if a particular host should be granted

access to a logical volume.

We highlight an embodiment of the inventive procedure 300 relative to the flow chart diagram of FIG. 3 (FIG. 3a and FIG. 3b) and which begins with a determination as to whether there has been an attempt by a host to login (Step 302). When a host login attempt is detected (Step 302), the **Controller** 106 searches for the WWN 107 of the host attempting the login in the Host WWN List 153 (Step 305). If the WWN of the **controller** attempting the login is found (Step 306), the position of the host's WWN 107 in the Host WWN List 153 is the Host Index 151. If the WWN is not found, the WWN 107 of the host attempting the login is added to the end of the Host WWN List 153 (Step 307) and that position is the Host Index 151. The Host Index 151 is then placed into the Host ID Map 155 at the position indicated by the host's **Fibre Channel** Loop ID 152 (Step 308). The **controller** 106 then collects the following information from the **Fibre Channel** I/O Processor 184: the **controller** (Step 309), the I/O Processor on which the request was made (Step 310), and the Logical volume for which the command was targeted (Step 311). (The process of collecting this information is typically unique to the particular hardware that implements the **Fibre Channel** I/O Processors 184, and therefore is not described here in detail.) This information allows the **controller** 106 to identify the correct Port Map Table 191 (Step 312), which contains the Volume Permission Table 194 for that logical volume 108. The **controller** 106 then searches the Volume Name Table 130 associated with that LUN to determine if that host attempting the login is allowed to access that logical volume 108 (Step 313). If a matching host WWN 107 is found in the Volume Name Table 130 for that logical volume 108, the **controller** 106 sets the Volume Permission Table 194 entry pointed to by the Host Index 151 to "true" or "1"(Step 315). If a matching WWN is not found for that logical volume 108, the **controller** 106 sets the Volume Permission Table 194 entry pointed to by HI to "false" or "0"(Step 314).

Controller 106 waits for a host access request (e.g. a command) to be received. On receipt of a host access request (for example, a read or write command, or an Inquiry or Vendor Unique command), controller **106 determines** the command type (Step 302). Once a command is received, controller **106 determines** the type of command to be an I/O command (such as a Read Command or a Write Command), or a Vendor Unique Command or Inquiry Command (Step 303).

If the request is an I/O Command (for example, a Read command, a Write command, or an Inquiry command), **controller** 106 determines the identity of the **controller** in which the command was received (Step 317), the host port of the command (Step 318), and the LUN and corresponding logical volume to which the command is addressed (Step 319). The proper Port Mapping Table is located based on the **controller,** host port I/O processor, and logical volume (Step 320); and the Host Index in the Host ID map is identified based on the Target ID of the command (Step 321). **Controller** 106 then examines the Volume Permission Table 194 at the position pointed to by the Host Index of the command to determine if the position stores a "1"bit (true) or a "0" bit (false) (Step 322). If the Permission Indicator value is true, access to the logical volume is permitted and **controller** 106 processes the command normally (Step 325). The process then completes and returns (Step 326). If the value is false, access to the logical volume is not permitted, **controller** 106 responds with an error condition (Step 324), such as an error condition indicating that storage space is not available for that logical volume, and the process completes and returns (Step 326). If the request is not an I/O command but instead either an Inquiry Command or a Vendor Unique command, then the response depends on the type of command. If the request is a Vendor Unique command, **controller** 106 processes the command normally, and returns (Step 304).

As already described, Host-to-Volume Mapping (HVM) is an enhancement and

extension of the Volume Mapping (VM) capability of the array **controller** already described, and maintains a port mapping data structure on a per logical volume basis. By "per logical volume basis" we mean that the port mapping data structure is instantiated for each logical volume. In the HVM enhancement we provide the host's WWN as a further access path qualifier.

The inventive procedure 300 is advantageously implemented as a computer program product 301 defined and stored in the memory, usually NVRAM 182, of **controller** 106 and optionally stored in memory of a host or on other storage media and downloadable to the **controller**. The program product 301, or executable portions thereof, is moved from memory 182 to RAM 181 associated with processor 180 of **controller** 106, and is executed by the Processor 180 within the **controller**. Processor Memory 181-182 refers to RAM, ROM, NVRAM and combinations thereof. Data to be sent between the host computer 101 and the logical volumes or disk drives 108 is buffered in the Data Cache Memory 186 which is accessed though the PCI Bus Interface and Memory **Controller** 183, though other interfaces may be used. The **Fiber Channel** I/O Processors 184 (184-1, 184-2, 184-3, . . . , 184-M) send and receive data from the host computers 101 and buffer this data in the Data Cache Memory 183. Likewise, the **Fibre Channel** or SCSI I/O Processors 185 (185-1, 185-2, . . . , 185-N) send and receive data from the logical volumes or disk drives 108 and buffer this data in the Data Cache Memory 186. Processor 180 coordinates the activities of all of the I/O processors 184-185, and handles scheduling of tasks including read and write tasks, and error handling.

The above described embodiments provide several advantageous features and capabilities. These include: (i) A Logical Volume maps to a single LUN only on a specific host port; (ii) a Logical Volume maps to the same LUN for all hosts that are granted access to the Logical Device in the Volume Permission Table (or Host Index Bit Map); (iii) a Logical Volume may map to different LUNs on a different **controller** or different host port; and (iv) multiple Logical Volumes may map to LUN 0 (or any other LUN) on a single host port, provided that there is no overlap of the Volume Permission Table (or Host Index Bit Map) for the Logical Devices.

We now return to a description of certain vendor unique commands so that the manner in which the system may be originally configured to accommodate HVM and reconfigured when changes or updates are desired, may be more readily understood. Vendor Unique commands allow the system 100 to be configured, and are not usually logical volume dependent. In this way, an array **controller** 106 that is not configured as part of the system 100 can be configured or re-configured to operate correctly with the unique WWNs 107 of the hosts 101 in the system 100. Configuring the **controller** to operate correctly with the logical volume 108 and with the unique storage requirements of the hosts 101 involves building a configurations data structure, and passing that data structure to the array **controller** through a Vendor Unique command.

We describe these logical volume configuration steps briefly. First, a user on the host computer (any of the host computers 101 connected to the **controller** 106) builds a configuration data structure in the hosts internal memory. (This process may also be automated based on information collected or available from other sources.) Next, the host computer transfers that configuration data structure to the array **controller** 106 though the Write Configuration variant of a Vendor Unique command. **Controller** 106 acknowledges the successful receipt of the command by returning a good SCSI status to the host in response to the Vendor Unique command. Fourth, at the completion of this Write Configuration Vendor Unique command, the array **controller** writes the configuration data to all of the disks (logical volumes) attached to the **controller**. Fifth, the host issues a Reset **Controller** Vendor Unique command to the array **controller** that

causes the **controller** to reset and restart. Finally, at the completion of restart, the **controller** 106 is configured as specified by the data in the configuration structure.

The earlier description also indicated that special conditions exist when the requested access is either an "Inquiry" command, a "Vendor Unique" command, or a "Report LUNs" command, as compared to a read or write command. An Inquiry command is a command that allows the host computer to determine if any data storage space is available for a specific SCSI Logical Unit and allows the host to determine the transfer characteristics for that SCSI logical unit. It returns specific information detailing the storage capacity of a SCSI LUN, the transfer capability of the LUN, serial numbers, and other information. A Vendor Unique command is a command that allows the unique characteristics of the array **controller** (for example, those characteristics not defined in the SCSI or **Fibre channel** specifications and therefore possibly not available via standard SCSI or **Fibre Channel** commands or protocols) to be determined and set as well as allowing other special operations to the storage array 108. This special treatment allows a **controller** that is not configured to be re-configured to operate correctly with the attached hosts. Examples of Vendor Unique type commands include the Set Configuration command and the Read Configuration command for reading and setting the array **controller's** configuration, and the Pass Through Operation command which allows the host direct access to the disk devices attached to the **controller**. These commands are known in the art and not described here in greater detail, except as necessary to describe special handing related to the invention.

If the request is an Inquiry command, the array **controller** 106 will return conventional Inquiry Data, and will indicate whether or not that host has access to the logical unit (and hence the logical volume. If the host does not have access to the logical volume, the **controller** will return the Inquiry Data with the Peripheral Qualifier set as an indicator to indicate that the array is capable of supporting a device on this SCSI logical unit, but that no device is currently connected to this SCSI logical unit.

Inquiry Commands are handled in this way in part because the SCSI specification states that a SCSI LUN should always return Inquiry Data. Inquiry data is status data about the SCSI device, and has nothing to do with data stored on that device. It is an issue for the command and the host to determine if the device has any storage, and to determine what the device is capable of, for example, how fast it can transfer data. The SCSI protocol runs on top of the **Fibre channel** layer, so this description is applicable to both parallel SCSI and **Fibre channel** implementations of the invention. Where conformity with the SCSI specification is not required, alternative procedures may be substituted.

Finally, if the request is a "Report LUNs" command, and the addressed LUN is 0 (LUN=0 is required by the SCSI specification), then the **controller** completes the command normally, reporting only the LUNs accessible by the host requesting the command. A Report LUNs command returns information that details which SCSI Logical Units are available to the host on that **fibre channel** at that SCSI address.

For other than Inquiry, Vendor Unique, and Report LUNs type commands, if a request is made by a host for a logical volume and the logical volume does not have permission for that host, the array **controller** will assert an error condition and deny access. For example, the error condition may be asserted by setting a SCSI Check Condition status for that command, and returning SCSI Sense Data with the Sense Key set to Illegal Command and the Additional Sense Code set to Logical Unit Not Supported. Check Condition, Illegal Command, Sense Data, Sense Key, Additional Sense Code, Peripheral Qualifier and Logical

Unit Not Supported are standard SCSI terms and are not described further here.

In addition to these commands, Host-to-Volume Mapping (HVM) advantageously uses several particular Vendor Unique direct commands. These are referred to here as Host-to-Volume Mapping (HVM) Direct Commands. A direct command is a SCSI Vendor Unique Command that allows configuration data to be sent and received by the array **controller**. These Host-to-Volume Mapping (HVM) Direct Commands include: Read LUN Map, Write LUN Map, and Read Volume WWN Table.

The Read LUN Map command returns to the host, Volume Mapping information maintained by array **controller** 106. The host needs Volume Mapping information from the **controller** in order to display the current configuration of the logical volume array to the user. The mapping information is stored in the logical volume Port Mapping Table data structure defined in the configuration data of the **controller**. This data is stored in the non-volatile memory of the array **controller** 106, and preferably in special reserved areas (COD) of the disk drive as well.

In one embodiment of the invention, the Read LUN Map command is sent using Vendor Unique Direct Command opcode (20h) in the **controller** firmware. An exemplary command format is illustrated in Table I.

In this exemplary Command Data Block (CDB) format, the LUN field contains the logical unit number of the CDB, and is ignored. The Direct Command Opcode (DCMP OP CODE) is the command to be executed, and MDACIOCTL_READLUNMAP (D1h) is the specific command value for the Read LUN Map command. The Logical Volume Number (Most Significant Bits--MSB and Least Significant Bits--LSB) specifies the device number of the Logical Volume whose information is to be reported. The Allocation Length (MSB and LSB) indicates the number of bytes the initiator has allocated for returned information. If the length is zero, no data is transferred and this is not treated as an error condition. The **controller** terminates the data phase when it has completed the transfer of the requested number of bytes or all returned Volume Mapping information, whichever is less. All Reserved fields and Control Byte (which is ignored here) should be 0.

Error conditions for the Read LUN Map command include standard SCSI responses for an error, including that an invalid Logical Volume number was specified. The **controller** will also respond to a SCSI Check Condition Status, such as will occur when a non-existent logical volume is specified in the command.

The Write LUN Map Vendor Unique Direct command allows an initiator, such as a host computer, to create or change the Host-to-Volume Mapping (HVM) information used by the **controller**. The Host-to-Volume Mapping (HVM) information is created when the **controller** is initially configured, and is changed when logical volumes are added or deleted, or when host computers are added or removed. This data format reflects the Port Mapping Table data structure. An exemplary WriteLUN Map Direct Command CDB Format is illustrated in Table II.

The operation code (DCMD OP CODE) field value for the write LUN map (MDACIOCTL_WRITELUNMAP) (D2h) specifies the direct command to write the LUN map. The Logical Volume Number specifies the device number of the logical device whose information is to be reported. The Allocation Length indicates the number of bytes the initiator is going to send to the **controller**. If the length is zero, no data is transferred and this is not treated as an error condition. All Reserved fields and Control Byte must be 0. Error conditions for the Write LUN Map include standard SCSI responses for an error, including that an invalid or non-existent Logical Volume number was specified.

The Read Volume WWN Table command returns the Volume WWN Table maintained by

the **controller**. The data returned by this command provides a translation from a host's physical WWN to the Host Index used internally by the **controller** and by the Read/Write LUN Map commands. This information is necessary when the host computer constructs the information for the configuration sent during a Write LUN Map command. An exemplary CDB for Read Volume WWN Table Vendor Unique Direct Command is illustrated in Table III.

The operation code field (DCMD OP CODE) labeled field read host WWN field (READ_HOST_WWN_TABLE) specifies the direct command to read the host WWN table. This command may be adapted to return a desired number of bytes of data per host supported. The number of bytes returned are usually determined by the particular host computer. It should be ready to accept the data the **controller** sends, so it needs to have enough memory space available to store the data. This may typically vary from computer to computer. For example in one embodiment of the invention, the command returns twelve bytes of data per host, while in another embodiment of the invention, the command returns 192 bytes of data per host supported, and in yet another embodiment of the invention, the firmware in which this command is implemented returns 3072 bytes of data per host supported.

External configuration programs such as GAM (Global Array Manager) or RAIDfx can use the data from the Read Volume WWN Table command to determine some limited information regarding the fibre host cabling topology. Hosts available for assignment in the HVM should be displayed by their respective WWN for **fibre channel** topologies. The concept of the Host Index may normally be hidden from the end-user, as the assignment of host indexes is arbitrary, with the Host Index having no fixed relation to the physical host. Once Host Indexes are assigned, they remain fixed until the configuration is cleared. A simple graphical representation of the host cabling and connection topology may optionally also be provided to the user to aid the end-user in properly determining a viable Host-to-Volume Mapping (HVM) strategy. External configuration programs may also be provided with a "probe" for attached storage through other hosts on a network to enable the configuration program to associate the actual network name of the attached hosts with their respective WWN. Translation and conversion procedures may optionally be provided for legacy systems and configurations that were implemented prior to HVM.

We now describe SCSI command support in the HVM environment and exemplary **controller** responses to commands in one embodiment of the invention when Host-to-Volume Mapping (HVM) is used in a standard SCSI command environment. The Host-to-Volume Mapping (HVM) feature limits access to Logical Volumes based upon the identity of the host requesting a command, and the specific command sent.

The **controller** always responds to a SCSI Inquiry command from any host and to any LUN with good status. If the host does not have access to the Logical Volume, the **controller** returns the Inquiry data with the Peripheral Qualifier set to indicate that the target is capable of supporting the specified device type on this LUN, but no device is currently connected to that LUN. If the host does have access to the Logical Volume, the **controller** returns its normal Inquiry data. The SCSI Report LUNs command is always supported on LUN 0, regardless of the host sending the command or the **controller** port the command was received on. The **controller** returns information only about the LUNs that the host requesting the command has access. For the SCSI Request Sense command, if a host does not have access to the addressed LUN, the **controller** returns sense data with the sense key set to Illegal Request and the additional sense code set to Logical Unit Not Supported. All other standard SCSI commands are terminated with Check Condition status and auto sense data containing a sense key set to Illegal Request and the additional sense code set to Logical

Unit Not Supported. Where a command operates on specific Logical Volume, such commands are generally are terminated with Check Condition status if the host does not have access to the addressed Logical Volume.

The inventive structure and method may also be used in an Internet configuration or with any interconnected network of host computer systems and/or devices such as wide area networks (WANs) and storage area networks (SANs). While the external communication net increases in speed, the storage area network speed stays about 10 times faster. Furthermore, while we describe a structure and method that is based upon the WWN of a **fiber channel** device, other unique identifiers may be used, for example the serial number that is imbedded in certain host computer processor chips, such as the Intel Pentium III microprocessor chips. These and other identifiers may alternatively be used. As the bandwidth of external nets (WANs) becomes comparable to the storage area nets (SANs), the structures, procedures, and methods described here may be implemented for distributed storage on the Internet or on other interconnected networks of host computers, storage devices, information appliances, and the like, much in the manner that web pages on the Internet are distributed and linked.

2. The method in claim 1, wherein said data structure is defined in a memory of a memory **controller** controlling said hardware device.

5. The method in claim 4, wherein said data structure is defined in a memory of a memory **controller** controlling said hardware device.

collecting, by the **controller,** information from a channel I/O processor to allow the **controller** to identify the correct port mapping table data structure which contains the volume permission table data structure for a logical volume for which a request by the host was targeted, said information including: the **controller,** the I/O Processor on which the request was made, and that logical volume;

searching, by the **controller,** the volume number table data structure associated with that logical volume to determine if that host attempting the login is allowed to access that logical volume; and

if the WWN of the host attempting the login is found in the volume number table data structure for that logical volume, setting by the **controller,** the volume permission table data structure entry pointed to by the host index to a first logical state; but if the WWN of the host attempting the login is not found for that logical volume, setting the volume permission table data structure entry pointed to by host index to a second logical state.

waiting, by the **controller,** for a host access request to be received;

determining, upon receipt of a host access request by the **controller,** the command type;

if the command type is an I/O command type, the **controller** determines the identity of the **controller** in which the command was received, the host port of the command, and the LUN and corresponding logical volume to which the command is addressed;

locating, the proper port mapping table data structure based on the identity of the **controller,** the host port I/O processor, and the logical volume;

examining, by the **controller,** the volume permission table data structure at the position pointed to by the Host Index of the command to determine if the volume

permission table data structure entry pointed to by the Host Index stores a entry having the first logical state or the second logical state; and

if the volume permission table data structure entry has a first logical state, permitting access to the logical volume and processing the command by the **controller** normally; and if the volume permission table data structure entry has the second logical value then denying access to the logical volume and responding to the command with an error indication.

(iii) a logical volume may map to different logical unit numbers on a different **controller** or different host port; and

12. The method in claim 4, wherein said hardware device comprises a RAID storage system and said **controller** comprises a RAID array **controller**.

(iii) a logical volume may map to different logical unit numbers on a different **controller** or different host port; and

a **controller** coupled between said plurality of host computers and said at least one shared hardware device and controlling access to said hardware device by said host computers; and

a data structure defined in a memory of said **controller** and comprising: (i) a host computer ID map data structure; (ii) a port mapping table data structure comprising a plurality of port mapping table entries; (iii) a host identifier list data structure in said memory; (iv) a volume permission table data structure; and (v) a volume number table data structure;

17. The interconnected network of computers in claim 14, wherein said communications channel comprises a **fibre channel** arbitrated loop communications channel.

said communications channel comprises a **fibre channel** arbitrated loop communications channel; and

21. A **controller** for controlling access to at least one shared hardware device that is coupled with a plurality of host computers by a communications channel and having a locally unique node identifier, said **controller** comprising:

22. The **controller** in claim 21, wherein said at least one shared hardware device comprises an information storage device.

23. The **controller** in claim 21, wherein said at least one shared hardware device comprises a logical volume of a disk drive storage subsystem.

24. The **controller** in claim 21, wherein said communications channel comprises a **fibre channel** arbitrated loop communications channel.

25. The **controller** in claim 21, wherein said locally unique node identifier comprises a world wide number (WWN) identifier.

26. The **controller** in claim 21, wherein said hardware device comprises a Storage Area Network.

27. The **controller** in claim 21, wherein:

said communications channel comprises a **fibre channel** arbitrated loop communications channel; and

28. The **controller** in claim 21, wherein said instructions include instructions for:

29. The **controller** in claim 28, wherein said instructions further include instructions for:

defining a data structure in a memory of a **controller** controlling said at least one shared hardware device, wherein defining comprises using said locally unique identifiers identifying which particular ones of said computers may be granted access to said device based on a logical configuration between said computers and said hardware device allowing one or more computers to access said hardware device and denying access to said hardware device by other of said computers, said data structure providing a configuration information that makes any particular logical volume visible to selected ones of said computers and invisible to other ones of said computers;

collecting, by the **controller,** information from a channel I/O processor to allow the **controller** to identify the correct port mapping table data structure which contains the volume permission table data structure for a logical volume for which a request by the host was targeted, said information including: the **controller,** the I/O Processor on which the request was made, and that logical volume;

searching, by the **controller,** the volume number table data structure associated with that logical volume to determine if that host attempting the login is allowed to access that logical volume; and

if the WWN of the host attempting the login is found in the volume number table data structure for that logical volume, setting by the **controller,** the volume permission table data structure entry pointed to by the host index to a first logical state; but if the WWN of the host attempting the login is not found for that logical volume, setting the volume permission table data structure entry pointed to by host index to a second logical state.

waiting, by the **controller,** for a host access request to be received;

determining, upon receipt of a host access request by the **controller,** the command type;

if the command type is an I/O command type, the **controller** determines the identity of the **controller** in which the command was received, the host port of the command, and the LUN and corresponding logical volume to which the command is addressed;

locating, the proper port mapping table data structure based on the identity of the **controller,** the host port I/O processor, and the logical volume;

examining, by the **controller,** the volume permission table data structure at the position pointed to by the Host Index of the command to determine if the volume permission table data structure entry pointed to by the Host Index stores a entry having the first logical state or the second logical state; and

if the volume permission table data structure entry has a first logical state, permitting access to the logical volume and processing the command by the **controller** normally; and if the volume permission table data structure entry has the second logical value then denying access to the logical volume and responding to the command with an error indication.

defining a data structure in a memory of a **controller** controlling said hardware

US-PAT-NO: 6449709

DOCUMENT-IDENTIFIER: US 6449709 B1

TITLE: Fast stack save and restore system and method

---------- KWIC ---------

A processor includes a stack that operates as a circular stack and appears to the address space in the memory of the processor as a single point address location. The stack supports read and write data access functions in addition to CALL (push) and RETURN (pop) programming operations. The processor may be programmed to save the stack in a typical manner with one instruction atomically transferring each element in the stack directly from the stack to a save storage. To restore the stack, the processor may be programmed to individually restore each element. The processor supports a special MOV instruction that transfers a plurality of bytes in a single operation. The special MOV instruction has one argument that **identifies** the beginning transfer source address, another argument defines the byte count indicating the number of bytes to be transferred, and a beginning transfer destination address. The processor may be programmed to perform a stack save operation with only a single instruction that moves the contents of the stack to the save storage. To further reduce context switching time and reduce the stack save and restore operation to a minimum number of instructions while maintaining the proper entry relationship for both stack read and write operations, the processor includes a "stack read forward" option to the special MOV instruction. The option to the special MOV instruction operates to read data in a forward direction even when no valid data is stored in the locations. The read operation begins at the start address specified by an argument to the MOV instruction, reads forward, and wraps around in a binary fashion back to the start address.

The processor supports a special MOV instruction that transfers a plurality of bytes in a single operation. The special MOV instruction has one argument that **identifies** the beginning transfer source address, another argument defines the byte count indicating the number of bytes to be transferred, and a beginning transfer destination address. The processor may be programmed to perform a stack save operation with only a single instruction that moves the contents of the stack to the save storage.

Advantageously, the Multi-Tasking Protocol Engine 250 supports a special MOV instruction that transfers a plurality of bytes in a single instruction. The special MOV instruction has one argument that **identifies** the beginning transfer source address, another argument defines the byte count indicating the number of bytes to be transferred, and a beginning transfer destination address. The Multi-Tasking Protocol Engine 250 may be advantageously programmed to perform a stack save operation with only a single instruction that moves the contents of the stack 480 to the SRAM memory 142. However, the Multi-Tasking Protocol Engine 250 includes source and destination addressing functionality that supports a fixed mode and an incrementing mode, but not a decrementing mode. Thus, the entry order that is stored in the SRAM 142 is the reverse order of the entry order read operation so that one technique for restoring the stack is to individually restore each element in the stack 480. Thus eight instructions are used to restore the stack 480 so that a complete stack save and restore operation that returns stack elements to the proper order includes a total of nine instructions.

FIG. 1 shows a computing system 100 in accordance with an embodiment of the invention. Computing system 100 includes a host computer 110, which has a system bus 120, and system bus devices 130 to 132 that are connected to system bus 120. Device 130 is a Fibre Channel controller integrated circuit (IC) component that includes a host adapter 140 for control of a peripheral bus 143 connected to a media interface serializer/deserializer (SERDES) 141 chipset to perform selectable parallel 20-bit or parallel 10-bit to serial high speed data transfers between a serial Fibre Channel (FC) loop 150 to FC device 160 and a parallel system Peripheral Component Interconnect (PCI) bus 120. The SERDES chipset performs parallel to serial send data conversion with internal high speed serial transmit clock generation, receive serial to parallel data conversion, receive word sync detection, receive data clock extraction, and serial data loopback functions. Host computer 110 can communicate via device 130, with devices 160, 170, and 180 that are connected to FC loop 150 and supports link module **identification,** attached media **identification,** and optical fiber safety sense and control. In particular, host computer 110 executes software including an operating system 112 and a device driver 114 for devices 160, 170, and 180. Device driver 130 includes a hardware interface module (HIM) 118 that communicates with device 130 via bus 120 and at least one upper layer module (ULM) 116 that communicates with devices 160, 170, and 180 via HIM 118 and device 130.

Host adapter 140 is a programmable integrated circuit that includes a multi-tasking protocol engine. The multi-tasking protocol engine executes software or firmware for controlling communications between host computer 110 and devices on bus 150. Coupled to host adapter 140 is a local memory including volatile memory 142 and non-volatile memory 144 and 148. Volatile memory 142, typically DRAM or SRAM and preferably a synchronous SRAM, is for information such as transfer control blocks for devices FC device 160, host system 180, and SCSI device 170. The non-volatile memory including a conventional EPROM, EEPROM or Flash memory 148, and an SEEPROM 144 for critical configuration information and non-critical information. In the exemplary embodiment, SEEPROM is a 1-Kbit memory that stores a world-wide port and node name, a local address, a subsystem-**ID,** a subsystem vendor **ID,** a preferred FC port address, external ROM/EEPROM size information, and other board related data. The world wide address is a world-wide unique address assigned to each port in the network and is represented as a 64-bit unsigned binary value. In addition, a 64-bit world wide node address is assigned to each node in a port. Also stored in SEEPROM 144 are the subsystem vendor **ID** and the subsystem board **ID,** represented as 16-bit binary values. An 8-bit preferred FC port address, which the address for device in an arbitrated loop, may also be stored in SEEPROM 144. U.S. Pat. No. 6,240,482 B1 issued on May 29, 2001, further describes the use and organization of the local memory space (e.g., memories 142, 144, and 146) of a host adapter and is hereby incorporated by reference in its entirety.

Multi-tasking protocol engine 250 executes protocol commands described by a Transfer Control Block (TCB) and scatter/gather (S/G) lists to control the data transfer between the host system memory and the **Fibre Channel** connected device. A TCB is a data structure that contains all information for the execution of a command. TCBs are prepared by the device driver in a host system memory TCB array along with the associated S/G elements. In the illustrative computing system 100, the **Fibre Channel** (FC) device 160 executes high-speed **Fibre Channel** protocol transfers with the Multi-Tasking Protocol Engine 250 performing initialization and **monitoring** functions. The Multi-Tasking Protocol Engine 250 handles **Fibre Channel** protocol transfers by executing operations based on a clock rate referenced to a **Fibre Channel** clock (not shown). Multi-tasking protocol engine 250 transfers TCBs from system memory to local memory 142 of host adapter 140 for access when host computer 110 indicates the TCBs are

available. Multi-tasking protocol engine 250 connects via an internal bus CIOBUS to memory port interface 230 which provides access to the local memory. Bus CIOBUS connects to multi-tasking protocol engine 250, memory port interface 230, FC data path 260, command management channel 220, and host interface 210. To access local memory, multi-tasking protocol engine 250 first acquires control of bus CIOBUS from a bus arbitrator (not shown). Multi-tasking protocol engine 250 can then read from local memory via memory port interface 230, from a buffer memory in command management channel 220, or from host interface 210. Host interface 210 or command management channel 220 can similarly acquire control of internal bus CIOBUS and access memory via memory port interface 230.

The Multi-Tasking Protocol Engine 250 uses the command management channel 220 DMA channel to transfer TCBs from the TCB array in the host system memory to a TCB synchronous SRAM array 142 connected to the memory port interface 230. The Multi-Tasking Protocol Engine 250 transfers the contents of a TCB to the appropriate registers for execution. The TCBs are executed independently of the Fibre Channel Target ID in the order received. The Multi-Tasking Protocol Engine 250 handles all normal protocol command activity with or without a host system interrupt upon command completion.

DINDIR register is an indirect address destination register for indirectly addressing the destination write register DINDEX. When a transfer is made to the destination, the contents of register DINDEX identify the destination address. The contents of register DINDEX are auto-incremented the clock cycle after DINDIR has been addressed except when DINDEX addresses a data port during a MOV instruction until the byte count expires.

The host computer 110 writes a value to sequencer RAM address (SEQADDR) register with the sequencer control (SEQCTL) register LOADRAM bit clear then restarts the multi-tasking protocol engine 250 by writing a 0 value to CMC host control register bits HPAUSETOP and HPAUSE. In response, instruction execution begins at the instruction identified by the value written by the host computer 110 and the value plus one is stored in the sequencer RAM address (SEQADDR) register.

In one example of the operation of the sequencer RAM address (SEQADDR) register, the multi-tasking protocol engine 250 writes a value to sequencer RAM address (SEQADDR) register with a MOV instruction that contains a byte count equal to two. In response, the instruction identified by the value is the next instruction to be performed and the value plus one is stored in the sequencer RAM address (SEQADDR) register.

In another example, the multi-tasking protocol engine 250 executes a RET instruction or an instruction containing an attached RET field. In response, the last value stored in the STACK register causes the instruction identified by the value to be the next instruction executed and the value plus one is stored in the sequencer RAM address (SEQADDR) register.

When a CALL instruction is executed, the value in the sequencer RAM address (SEQADDR) register is pushed onto the STACK register. The instruction identified by the value in the next address field of the CALL instruction is the next instruction to be executed, and the value plus one is stored in the sequencer RAM address (SEQADDR) register.

For execution of a JUM, JC, JNC, JZ, or JNZ instruction in which the jump or branch is the action executed, then the instruction identified by the next address field of the instruction is the next instruction to be executed. The next address field value plus one is stored in the sequencer RAM address (SEQADDR) register.

When a 1 is written to a SEQRESET bit of a sequencer control[1] register, the value in the sequencer RAM address (SEQADDR) register is cleared to zero. The instruction **identified** by zero is the next instruction to be executed and one is stored in the sequencer RAM address (SEQADDR) register.

If an external firmware load control function, which is initiated by asserting an EXFLR signal, is completed with bit EXFLR_ADR_START clear, then sequencer RAM address (SEQADDR) register is cleared to zero. The instruction **identified** by the zero value is the next instruction to be executed, and the next instruction value plus one is stored in the sequencer RAM address (SEQADDR) register. If an external firmware load control function is completed with bit EXFLR_ADR_START set, the instruction **identified** by the value in an external load address[9:0] register is the next instruction to be executed and the next instruction value plus one is written to sequencer RAM address (SEQADDR) register.

US-PAT-NO: 6381642

DOCUMENT-IDENTIFIER: US 6381642 B1

TITLE: In-band method and apparatus for reporting operational statistics relative to the ports of a fibre channel switch

---------- KWIC ---------

An in-band method/apparatus whereby a host is enabled to secure predetermined operational information relative to predetermined ports of a **fiber channel** switch. A set command is generated at the host and sent in-band to the switch. The information content of the set command defines the ports for which operational-parameters are to be **monitored**. The information content of the set command also defines which operational parameters are to be **monitored**. In response to receiving the set command, the switch establishes statistical counters for **monitoring** port operational parameters in accordance with received operational parameter **identifiers**. An accept signal is then sent in-band to the host, and a time period of port **monitoring** begins. After a predefined time period has expired, the host sends a read command in-band to the switch. The switch now generates a **monitor** record in accordance with the count content of the statistical counters that were established in response to the set command. This **monitor** record contains one port field for each of the ports that were specified by the set command, and each of the port fields contains one or more count fields that contain port operational count data for the port operational parameters that were specified by the set command. The **monitor** record is then sent in-band to the host. Recycling of the timed process may be provided.

In accordance with this invention, a host or host client specifies the format of set monitor commands, or monitor requests that are provided to a management director that is within the FC switch. This format allows a host client program to (1) specify the number of port counters that are to be monitored, (2) **identify** whether external ports or an internal port are to be monitored, and (3) specifically **identify** the ports that are to be monitored.

In accordance with this invention, the format of monitor information records or monitor responses that are returned from the FC switch to the host client is also defined. This monitor response format specifies a sequence counting scheme that guarantees the delivery of port statistics that provide a mechanism to concatenate multiple monitor records, and that provides an **identification** of each multiple monitor record that is transmitted to the host client.

In accordance with the invention, management director 24 generates statistical information concerning the performance of **fibre channel** links 17 that connect internal port 16 to the various F_ports 11. This statistical information is presented to host client 13 in the format of a **Monitor** Information Record 45 (see FIG. 4). This **Monitor** Information Record 45 is generated as a result of a Set **Monitor** Command 39 (see FIG. 3), and a subsequent Read Port Statistics Command 41 (see FIG. 3) that are received by management director 24 from host client 13.

Statistical counters 18 are used to provide specified pieces of port-operational information that relate to the performance of each F_port 11, or to the performance of preselected F_ports 11, and/or to the performance of internal port 16. This counter information is read by virtue of host client 13

issuing an in-band Read Port Statistics Command 41 at a time that is subsequent to host client 13 issuing an in-band Set Monitor Command 39 that specifies which statistical counters 18 are to be established/read. Statistical counters 18 are reset to zero by a power on reset of FC switch 10, and may also be reset to zero by an internal reconfiguration of FC switch 10. The information content of each one of the statistical counters 18 is **identified** by, or is related to, a statistical counter **identifier or id** code (see TABLE-1) in accordance with the port parameter information that is contained therein.

In an embodiment of the invention, the statistical counter **identifier** (contained in a counter control word of set command 39) for an individual counter 18 comprises a 2-byte hexadecimal code as shown in the following TABLE-1.

As a nonlimiting and simplified example, the content of this Set Monitor Command signal 39 may comprise "present external port information, start port=5, end port=7; words transmitted=0903; words received=0904". This example content of a Set Monitor Command signal 39 operates to specify that the statistical port counters 18 of F_ports 5 through 7, inclusive, are to be monitored, that the quantity of the words transmitted by an F_port is to be **identified by the identifier** "0903", and that the quantity of words received by an F_port is to be **identified by the identifier** "0904". Without limitation thereto, in this embodiment of the invention, the word transmitted and the words received are to be periodically monitored relative to a 5-second time interval that is measured by function 33.

(6) a number of counter control words that each contain a TABLE 1 parameters to be counted, and that **identify** the counter control word that is last operative counter control word.

"count provided, not last; internal port, port 5; count **id**=0903, count statistics=173490; count **id**=0904, count statistics=88551"

"count provided, not last; internal port, port 6; count **id**=0903, count statistics=90123; count **id**=0904, count statistics=721183"

"count provided, last; internal port, port 7; count **id**=0903, count statistics=213; count **id**=0904, count statistics=1276".

Also note that in accordance with the format of Monitor Information Record 45, each individual portion of Monitor Information Record 45 indicates the related port number,. wherein count statistics that indicate the number of words transmitted by the related port is **identified by the identifier** "0903", and wherein count statistics that indicate the number of words received by the related port is **identified by the identifier** "0904".

The format of this command is shown below in TABLE 2, and comprises the quantity 64 of 4-byte words, made up of word 0 through word 63. As can be seen, byte 3 of word 2 specifies a starting port number, byte 3 of word 3 specifies an ending port number, and words 4 through 63 provide the quantity 60 of counter control words that are **identified** as counter control words 0 through 59. Counter control words 0 through 59 are provided for use in specifying port operational parameters (see TABLE 1) that are to be monitored or counted.

Bit 0 provides external port information, and indicates whether statistical information is to be presented for one or more external F_ports. When bit 0 is set to 0, statistical information will not be presented for any external F_port, and the starting port/ending port (byte 3 fields of words 2 and 3) is ignored. When bit 0 is set to 1, statistical information will be presented for

the external F_ports that are defined, or **identified** by the starting port/ending port byte 3 fields of words 2 and 3.

Bit 2 is a counter set bit that indicates whether the invention is to report on all statistical counters 18, or is to report on only the counters 18 that are **id**-specified by counter control words 0 through 59. When bit 2 is set to 0, the invention monitor will report on only the statistical counters 18 that are **id**-specified by counter control words 0 through 59. When bit 2 is set to 1, the invention will report on all statistical counters 18 to subsequently occurring Read Port Statistics Commands 41 (issued in-band by host client 13), in which case, any counter control words contained in words 4 through 63 are ignored.

Note that the above-mentioned port numbers are physical port numbers that **identify** physical F_ports, and statistical information is presented in association with these physical F_port numbers rather than in association with port addresses.

Words 4 through 63 of the Set Monitor Command (i.e., the counter control words) each contain one 32-bit counter control word wherein bits 0 through 7 contain a counter control field, bits 8 through 15 are reserved, and bits 16 through 31 comprise a statistical counter **identifier** or a counter **id** as shown in TABLE 1.

Bits 16 through 31 of a counter control word contain a 2-byte code that contains the statistical counter **id** of a counter that is to be included in the information that is read in-band as a result of subsequent Read Port Statistics Commands 41 that are issued in-band by host client 13. This counter **id** code specifies the port operational parameter that is to be monitored by the related counter (see TABLE 1).

The flag field of a monitor header record 46 (i.e., byte 0 of word 0) **identifies** the format of the information that is contained in the remainder of the record. For a monitor header record 46 this flag is set to hexadecimal

A monitor port information record 49, 53 immediately precedes the set of counter statistical records for a given port, as shown in FIG. 4. This monitor port information record 49, 53 **identifies** the port, and provides information concerning the state of the port. The following TABLE 4 shows the format of a monitor port information record.

The flag field, i.e. byte 0 of word 0, **identifies** the format of the information that is contained in the remainder of the monitor port information record. For a monitor port information record, this flag field contains hexadecimal "61".

The port descriptor field, i.e. word 1, is defined as follows: When bit 2 of the status field (i.e., bit 2 of byte 2 of word 0) is a "0", then the port descriptor field (i.e., word 1) contains the 32-bit port descriptor for the port that is **identified** in the port number field (i.e., in byte 3 of word 2). When bit 2 of the status field (i.e., bit 2 of byte 2 of word 0) is a "1", then the port descriptor field (i.e., word 1) contains all zeros.

Bit 2 indicates whether the port **identified** by the port number (i.e., by byte 3 of word 2) has an assigned port address. When bit 2 is set to "0", a port address assignment exists, and word 1 contains the port descriptor of the port. When bit 2 is set to "1", a port address assignment does not exist, and word 1 contains all zeros.

Bit 3 indicates whether this monitor port information record **identifies** an external port or an internal port. When bit 3 is set to "0", the current

monitor port information record describes an external port. When bit 3 is set to "1", the monitor port information record describes an internal port. In either case, the port number field (i.e., byte 3 of word 2) contains the port number that **identifies** the external/internal port.

A device end status of management director 24 is presented when management director 24 is ready to accept a command to which a command retry status had previously been presented. A device end status is **identified** by a status field in which only the device end bit is set. A device end status is also presented after requested data has become available following the presentation of a command retry status.

More specifically, Set Monitor Command 39 is generated at host client 13 and then sent in-band to FC switch 10. The information content of Set Monitor Command 39 (TABLE 2) specifically defines the ports of FC switch 10 for which operational parameters are to be monitored or counted (TABLE 2, bytes 3 of words 2 and 3). In addition, the information content of Set Monitor Command 39 specifically defines which operational parameters are to be monitored by way of 2-byte code counter **identifiers** (TABLE 1) that are contained in counter control words 0 through 59 of set command 39 (TABLE 2).

This Set Monitor Command 39 (TABLE 2) is received by FC switch 10. In response to receiving Set Monitor Command 39, switch 10 establishes the requested statistical counters 18 that are to monitor port operational parameters in accordance with the received ones of the counter **identifiers** (TABLE 1).

FC switch 10 now operates to generate a Monitor Information Record 45 (FIG. 4) in accordance with the count content of the statistical counters 18 that were established in response to Set Monitor Command 39. This Monitor Information Record 45 contains one field (47, 48, etc.) for each of the ports that were specified by Set Monitor Command 39, and each of the port fields contains a count field (50-52, etc. and 54-56, etc.) that contains port-operational count data for the port-operational parameters that were specified by the counter **identifiers** contained in Set Monitor Command 39.

TABLE 1 **ID** CODE PORT OPERATIONAL PARAMETER 09 01 Number of words transmitted 09 02 Number of words received 09 03 Number of frames transmitted 09 04 Number of frames received 09 05 Number of class 2 frames received 09 06 Number of class 3 frames received 09 07 Number of link control frames received (FC link frames) 09 08 Number of multicast frames received 09 09 Frame pacing limit (Number of 2.5 microsecond units that frame transmission is blocked due to zero credit) 09 10 Number of disparity errors in frames 09 11 Number of CRC errors 09 12 Number of frames greater-than FC maximum 09 13 Number of frames less than FC minimum 09 14 Number of frames with bad or missing EOF 09 15 Number of disparity errors outside of frames 09 16 Number of invalid or unrecognizable ordered sets outside of frames 09 17 Number of class 3 frames discarded 09 20 Number of link failures 09 21 Number of loss of synchronization detected by port 09 22 Number of loss of signal detected by port 09 23 Number of primitive sequence protocol 09 24 Number of invalid transmission words 09 25 Number of address **Id** errors 09 26 Number of LRR issued by port 09 27 Number of OLS received by port 09 28 Number of OLS issued by port

4. Apparatus enabling a host computer to **monitor** specified port operating parameters relating to specified ports of a **fibre channel** switch, comprising:

US-PAT-NO: 6367033

DOCUMENT-IDENTIFIER: US 6367033 B1

TITLE: Method and apparatus for recreating fiber channel traffic

---------- KWIC ---------

Various types of sophisticated analyzers have been designed to enhance the functionality of a logic analyzer. For example, U.S. Pat. No. 5,457,694 describes a bus analyzer used for an advanced technology attachment (ATA) bus. The ATA bus is also known as an integrated device electronics (**IDE**) bus. The ATA bus analyzer disclosed therein performs tasks related to trouble shooting and performance measurement. This system captures data from an ATA bus much like a logic analyzer. A trigger may be used to control the starting and stopping of data capture. This analyzer uses a filter function to throw away large volumes of useless information which otherwise require storage space and obscure the trouble shooting process. Also, this analyzer formats data and provides a menu driven user interface. This user interface allows the user to search through a database of captured data to locate a particular event detected on the ATA bus. However, this system is useful only for trouble shooting a source computer system directly. This system does not provide a means for data to be captured from a given source computer system and then replayed in on a reference system to duplicate a run-time error.

Another aspect of the present invention provides a test apparatus which may be used to evaluate problems **identified** remotely. This test equipment includes a host adapter coupled to an interconnect. The test equipment also includes a control module coupled to the host adapter and operative to convert a template data structure into a physical layer data transfer. The physical layer data transfer takes place via the interconnect. The control module is also operative to record information related to a data transfer generated by a target device coupled to the interconnect. The template data structure is constructed to replicate a data transfer captured from a source computer system distinct from the apparatus.

In an embodiment of computer system 100 involving a very simple type of host-side hub 110, a point-to-point connection exists between a single host computer 105 and a single disk array controller such as disk array controller 115. In this case host-side hub 110 may involve a tapped connection from a **Fibre Channel** point-to-point link. This tapped connection is used to send a copy of data transferred on the point-to-point link to a host-side **monitor** and analyzer 125. Host-side **monitor** and analyzer 125 is coupled to a mass storage database 130. In a **Fibre Channel** arbitrated loop topology, host-side hub 110 may involve a node capable of routing traffic to host-side **monitor** and analyzer 125. In other embodiments, host-side **monitor** and analyzer 125 may be connected as a node into the arbitrated loop itself in which case function of the host-side hub 110 is built into host-side **monitor** and analyzer 125. In **Fibre Channel** systems involving a switching fabric oriented topology, host-side hub 110 may involve a switching fabric used to route traffic between nodes and send a copy of selected traffic to host-side **monitor** and analyzer 125. In some systems, the functionality of host-side **monitor** and analyzer 125 may be built into one of host computer 105's host adapters and the mass storage database 130 may be implemented on a hard disk connected into the system 100. While the foregoing illustrative embodiment describes a preferred configuration built

around the **Fibre Channel** Standard, the host-side data transfer mechanism may follow bus or network protocols other than **Fibre Channel**.

In embodiments involving any of the aforementioned topologies, host-side **monitor** and analyzer 125 may optionally be replaced by backside **monitor** and analyzer 140. Backside **monitor** and analyzer 140 is coupled to one or more disks as connected on the backside of disk array controllers 115 and 120. The backside involves signals sent between disk array controllers 115 and 120 and selected disks within the disk arrays 117 or 122. In systems involving multiple parallel backside connections such as disk array 117 which includes a plurality of parallel SCSI channels, a selector/concentrator 135 may be employed to select and route various signals to backside **monitor** and analyzer 140. Backside **monitor** and analyzer 140 is thus coupled to receive information from disk array 117 using selector/concentrator 135. In the illustrative embodiment of system 100, disk array 122 is preferably connected on the backside using a **Fibre Channel** arbitrated loop. Backside **monitor** and analyzer 140 is thus implemented with a **Fibre Channel** arbitrated loop interface to capture traffic sent across the loop.

In operation, computer system 100 executes one or more host programs, which give rise to a first data traffic flow observable from host-side **monitor** and analyzer 125. This first traffic flow involves data transfers between host computers 105 and other devices such as disk array controllers 115 and 120. These data transfers take place over a host-side interconnect which may preferably be implemented using a **Fibre Channel** compliant means as discussed above. The execution of one or more host programs also gives rise to a second data traffic flow observable from backside **monitor** and analyzer 140. In the illustrative embodiment, this second data traffic flow takes place between disk array controllers 115, 120 and disk arrays 117 and 122. Selector/concentrator 135 is operative to select channels on which data is to be transferred from parallel disk array 117 to backside **monitor** and analyzer 140. If backside disk array is configured into an arbitrated loop, backside **monitor** and analyzer 140 is preferably connected into the arbitrated loop and is operative to record selected information therefrom. Other topologies such as topologies involving a backside switching fabric or hub may be similarly constructed to route specified information from the backside data channels to backside **monitor** and analyzer 140. As discussed earlier, one or both of host-side **monitor** and analyzer 125 and backside **monitor** and analyzer 140 may be employed in a specific embodiment of system 100.

Data parser 410 involves a decoder portion, which is operative log information related to bus arbitration signals associated with a given data transfer into the template. The bus arbitration signals associated with a given data transfer are generated during an arbitration phase 415. Data parser 410 detects bus arbitration signals and logs into the template data structure an initiator **ID (identifier)** 417 associated with a host adapter which corresponds to one of the hosts 105. The initiator **ID identifies** the host which wins the arbitration and is thereby associated with the data transfer to follow. The decoder portion of the data parser is also operative to log information related to a selection phase 420 which follows the arbitration phase. In selection phase 420, a target **ID** 422 is extracted and logged into the template data structure. Target **ID** 422 **identifies** the target device with which the host adapter wishes to communicate. For example, the target device may correspond to disk array controller 115. A "source target device" is defined as a target device in source computer system 100. As discussed below, a target device in the reference system is used to replicate the actions of the source target device in the reference system.

Next a message portion of the data parser is operative to log information

related to a message phase 425 where a logical unit ID 427 is provided and a link layer data transfer protocol is negotiated. The message portion of the data parser logs the logical unit ID (also known as a LUN) into the template. The LUN designates a logical partition such as one constructed as a collection of sectors taken from each of the disks one through n in disk array 117. The message portion of the data parser is also operative to log information related to a link layer protocol negotiation 429 into the template. Unlike the previous information logged into the template, link layer protocol negotiation 429 involves a bi-directional data transfer whereby the host adapter associated with initiator ID 417 exchanges data with the target device associated with target ID 422.

Any or all of the aforementioned information fields (417, 422, 427, 429, 432, 437, 442) related to the template data structure are presented to a data interpreter/organizer 445. Data interpreter/organizer 445 receives template data structure information related to data transfers associated with an entire segment of a data-capture recording. Data interpreter/organizer 445 preferably filters useless information from the set of template data structures. For example, in some cases only data transfers involving a particular target ID may be of interest. If this is the case, all of the template data structures not involving this target ID may be discarded. The data interpreter/organizer may also associate time stamp information with each stored template data structure or provide other services to organize the template data structure data structures for use in playback and performance analysis. In most embodiments, data interpreter/organizer 445 outputs a stream which is placed into an output file or piped to another module such as a macro-level generator 450 or a performance analysis module 470. In cases where data interpreter/organizer 445 produces an output file, this output file will typically be used as input to macro-level generator 450 or performance analysis module 470.

In some systems, data interpreter/organizer 445 may alter data contained within a given template. For example, in source computer system 100 the target ID of disk array controller 115 has a first value, while in the reference system, an identical disk array controller has a different target ID. In such a case, the data interpreter/organizer is operative to translate the first target ID as used in source computer system 100 to the second target ID as used in the reference system. In general, data organizer 445 is operative to scan through the data produced by data parser 410 and format it to be executed on the reference system and/or analyzed.

The foregoing discussion also enables a test apparatus, which may be used to evaluate problems identified remotely. This test apparatus includes a host adapter coupled to an interconnect. The test equipment also includes a control module coupled to the host adapter and operative to convert a template data structure into a physical layer data transfer. The physical layer data transfer takes place via the interconnect. The control module is also operative to record information related to a data transfer generated by a target device coupled to the interconnect. The template data structure is constructed to replicate a data transfer captured from a source computer system distinct from the apparatus. In other words, the test apparatus according to the present invention has a structure similar to the aforementioned reference system, but may be used to connect directly to a device under test such as disk array controller 115 and disk array 117. The test apparatus may include host-side hub 110, or may be connectable thereto.

With reference now to FIG. 5, an example of a template data structure 500 is illustrated. In this example, the template data structure 500 tabulates information extracted from the data parser 410 as discussed in connection with FIG. 4. A first field 505 stores the initiator ID as extracted by data parser

410 in extraction 417. A second field 510 stores the target ID as extracted by data parser 410 in extraction 422. A third field 515 stores the logical unit ID (LUN) as extracted by data parser 410 in extraction 427. A fourth field 520 stores input and output negotiation information respectively in a subfield 521 and a subfield 522. These subfields hold the information extracted by data parser 410 in extraction 429. A fifth field 525 stores opcode, transfer length, and address information respectively in a subfield 526 a subfield 527 and a subfield 528. These subfields hold the information extracted by data parser 410 in extraction 432. A sixth field 530 stores the data payload as extracted by data parser 410 in extraction 437. A seventh field 535 stores the returned status information as extracted by data parser 410 in extraction 442. An eighth field 540 stores the statistical information such as a time-stamp which may be associated with a given data transfer. Other information including pointers to a next template data structure or other auxiliary information may be added to the template data structure. The template data structure may be defined in various ways depending on the programming language used to implement the processes of the present invention.

US-PAT-NO: 5901280

DOCUMENT-IDENTIFIER: US 5901280 A

TITLE: Transmission monitoring and controlling apparatus and a transmission monitoring and controlling method

---------- KWIC ---------

A transmission **monitoring** and controlling apparatus is provided between a first system and a second system. The transmission **monitoring** and controlling apparatus **monitors** information transmitted through a **Fibre Channel**. Then, a back controller analyzes the **monitored** information and stores the information in a controlling memory based on the analysis. For example, when the first system writes data on the second system, the back controller stores the data in the controlling memory. Therefore, even if a failure has occurred in the second system, the transmission **monitoring** and controlling apparatus is able to read the data from the controlling memory and send the data to the first system instead of to the second system.

Further, a transmission **monitoring** and controlling apparatus 2 is provided between the first system 1 and the second system 13. The transmission **monitoring** and controlling apparatus 2 **monitors** information which is transmitted through **Fibre Channel** 15, and controls the first system 1 or the second system 13 based on the **monitoring** result. The transmission **monitoring** and controlling apparatus 2 includes a connector 2a on a side of the first system 1 and a connector 2b on a side of the second system 13, each of which is connected to **Fibre Channel** 15.

The **monitor** .cndot. switch 3 **monitors** information which is transmitted through **Fibre Channel** 15. The **monitor** .cndot. switch 3 receives the information which is transmitted through **Fibre Channel** 15, and outputs the received information thoroughly. The **monitor** .cndot. switch 3 receives a command which is generated by the back controller 4, and switches so that the command does not conflict with other information which is transmitted through **Fibre Channel** 15. Then, the **monitor** .cndot. switch 3 sends the command to the first system 1 and the second system 13.

Besides, the magnetic disks 7-11 are connected to the first system 1 via the transmission **monitoring** and controlling apparatus and **Fibre Channel** 15. Further, the **monitor** .cndot. switch 3 includes a **monitor** 3a, a switch 3b, a **monitor** 3c, and a switch 3d. The **monitor** 3a **monitors** the information which is outputted from the transmitter (TX), and the **monitor** 3c **monitors** information which is outputted from the second system 13.

The receiving controller 21a receives the information which is **monitored by the monitor** 3a, and sends the information to the converter 23a. The receiving controller 21b also receives the information which is **monitored by the monitor** 3c, and sends the information to the converter 23b. Since the information which is transmitted through **Fibre Channel** 15 has a unit of ten bits, the converters 23a and 23b convert the information which is transmitted from the receiving controllers 21a and 21b to a unit of eight bits, and sends the converted information to the processing controller 22.

The **monitor** 3a receives the information from the transmitting controller 20a

and the information which is outputted from the first system 1, and the switch 3b controls sending of the information from the first system 1 to **Fibre Channel** 15 with a priority.

The transmitting controller 20b sends the received information to the **monitor** 3c. The **monitor** 3c receives the information from the transmitting controller 20b and the information from the second system 13, and the switch 3d controls sending of the information from the second system 13 to **Fibre Channel** 15 with a priority.

Owing to the operations of the switches 3b and 3d, the operation of the transmission **monitoring** and controlling apparatus 2 does not influence **Fibre Channel** 15. The operation of the transmission **monitoring** and controlling apparatus 2 is not related to **Fibre Channel** 15.

The information of the frame header includes routing data, a transmission destination port **ID**, control reserve, a transmission origin port **ID**, a protocol type, a frame generation control bit, a sequence **ID**, data field control, a frame number, an exchange **ID** on a transmitting side, an exchange **ID** on a responding side, and a relative offset.

In the transmission **monitoring** and controlling apparatus 2, the **monitor** 3a **monitors** the command to read data from the magnetic disk 10. The processing controller 22 analyzes the **monitored** information and realizes that the information is a command to read from the magnetic disk which has failed. Then, the processing controller 22 retrieves the data from the memory 5 or the magnetic disk 6. The processing controller 22 generates a command to send the retrieved data back to the first system 1, and outputs the command to **Fibre Channel** 15. The processing controller 22 changes a transmission origin port **ID** of the retrieved data to an **ID** of the magnetic disk 10 and sends the data to the first system 1, as if the data were retrieved from the magnetic disk 10.

A case in which a failure has occurred in the magnetic disk 10 is explained. If a failure has occurred in **Fibre Channel** 15 connected to the second system 13, the transmission **monitoring** and controlling apparatus 2 replaces the second system and writes or reads the data in response to a command from the first system 1.

For example, when the first system 1 executes a command to write data in the RAID configuration 14, the **monitor** 3a **monitors** the command to write and the processing controller 22 recognizes the **monitored** command to write. Then, the data are stored in the memory 5 or the magnetic disk 6. Location information for writing the data on the magnetic disk 8 is also stored. At this time, the command to write the data on the magnetic disk 8 is executed without any influence of the transmission **monitoring** and controlling apparatus 2. The processing controller 22 checks the vacancy of lines in predetermined constant intervals. When the processing controller 22 finds a vacancy in the lines, the processing controller 22 retrieves location information for writing from the memory 5 or the magnetic disk 6. The processing controller 22 generates a command to write the data on the magnetic disk 7 based on the location information for writing. Then, the processing controller 22 outputs the generated command to **Fibre Channel** 15.

Even when a failure has occurred in the magnetic disk 8, the transmission **monitoring** and controlling apparatus 2 stores the data and the location information for writing in the memory 5 or the magnetic disk 6 as if a failure had not occurred. When there is a vacancy in the lines, the transmission **monitoring** and controlling apparatus 2 generates a command to write the data on the magnetic disk 7, and outputs the generated command to **Fibre Channel** 15.

When the magnetic disk 8 has recovered from the failure, the transmission **monitoring** and controlling apparatus 2 also generates a command to write the data on the magnetic disk 8, and outputs the generated command to **Fibre Channel** 15.

In case the first system 1 attempts to execute a command to read data from the magnetic disk 8 which has failed, the transmission **monitoring** and controlling apparatus 2 retrieves the data from the memory 5 or the magnetic disk 6 instead of the magnetic disk 8 using the processing controller 22. Then, the transmission **monitoring** and controlling apparatus 2 generates a packet so that the first system 1 determines that the response is from the magnetic disk 8, and outputs the packet to **Fibre Channel** 15.

Normally, when it is impossible to retrieve data from the particular magnetic disk because the magnetic disk or a network has failed, redundant data are used to regenerate necessary data. Therefore, in embodiment 3, when the redundant data are to be written on the magnetic disk 7, the redundant data are not updated. When lines are open, the transmission **monitoring** and controlling apparatus 2 generates the redundant data, and outputs a command to write the redundant data on the magnetic disk 7 to **Fibre Channel** 15.

However, in embodiment 3, when the lines are open, the transmission **monitoring** and controlling apparatus 2 generates the redundant data. The transmission **monitoring** and controlling apparatus 2 also generates a command to write the redundant data on the magnetic disk 7, and outputs the command to **Fibre Channel** 15.

When the first system 1 generates commands to write in the RAID configuration 14 sequentially as shown in FIG. 10, the **monitor** 3a **monitors** the commands. Then, the processing controller 22 analyzes the information, and stores the redundant data updating information and the location information for writing in the memory 5 or the magnetic disk 6 as shown in FIG. 10. When the processing controller 22 recognizes a vacancy in the lines in **Fibre Channel** 15, the processing controller 22 checks the redundant data updating information sequentially in the order in which it is stored in the memory 5 or the magnetic disk 6. When the redundant data are not updated, the processing controller 22 generates the redundant data and a command to write the redundant data on the magnetic disk 7. Then, the processing controller 22 outputs the generated command to **Fibre Channel** 15.

The **monitor** .cndot. switch 30 in FIG. 14 differs from the **monitor** .cndot. switch 30 in FIG. 1 in its configuration. When the **monitor** .cndot. switch 30 in FIG. 14 receives information from the first system 1 or the second system 13, the **monitor** .cndot. switch 30 stores the received information without outputting the information to **Fibre Channel** 15. At the same time, the **monitor** .cndot. switch 30 sends the information to the processing controller 22. The processing controller 22 analyzes the information, and generates a command for the first system 1 or the second system 13 based on the analysis. Or, the processing controller 22 instructs the switch 3b or 3d to output the information which is stored by the **monitor** .cndot. switch 30 to **Fibre Channel** 15 thoroughly. The switches 3b and 3d switch the information which is stored by the **monitor** .cndot. switch 30 and the command which is generated by the processing controller 22 in response to an instruction from the processing controller 22, so that one of them is outputted to **Fibre Channel** 15.

Therefore, when the first system 1 sends a command to write data on the magnetic disk 9, the **monitor** .cndot. switch 30 stores the command to write the data, and sends the command to the processing controller 22. When the

processing controller 22 has recognized that the magnetic disk 8 is in a maintenance operation, the processing controller 22 stores the data to be written on the magnetic disk 9 in the memory 5 or the magnetic disk 6 instead. The processing controller 22 generates a response for the first system 1 as if the data are written on the magnetic disk 9, and sends the response to the **monitor** .cndot. switch 30. The **monitor** .cndot. switch 30 switches so that the response from the processing controller 22 is outputted to **Fibre Channel** 15 with priority. The command to write stored in the **monitor** .cndot. switch 30 is discarded. If the magnetic disk 8 is not in a maintenance operation and is in normal operation, the processing controller 22 instructs output of the command to write stored in the **monitor** .cndot. switch 30. Therefore, the command to write stored in the **monitor** .cndot. switch 30 is outputted to **Fibre Channel** 15.

However, in embodiment 4, when the lines are open, the transmission **monitoring** and controlling apparatus 2 generates the redundant data. The transmission **monitoring** and controlling apparatus 2 generates a command to write the redundant data on the magnetic disk 7, and outputs the command to **Fibre Channel** 15. Hence, the driver 16 in the first system 1 does not generate the redundant data. In embodiment 4, a part of the function of the controller is provided in the processing controller 22.

However, since the magnetic disk 8 is in the maintenance operation, the data cannot be written on the magnetic disk 8. Therefore, when the transmission **monitoring** and controlling apparatus 2 recognizes that the maintenance operation of the magnetic disk 8 is completed and there is a vacancy in the lines of **Fibre Channel** 15, the transmission **monitoring** and controlling apparatus 2 writes the data which are stored in the memory 5 or the magnetic disk 6 on the magnetic disk 8.

When the first system 1 outputs a command to read the data from the magnetic disk 8 during the maintenance operation of the magnetic disk 8, the **monitor** 3a **monitors** the command. The processing controller 22 analyzes the **monitored** command, and recognizes the command to read. The processing controller 22 obtains the data in reference to the information stored in the memory 5 or the magnetic disk 6 as shown in FIG. 13 based on the disk number and the address. Then, the processing controller 22 changes a transmission origin port **ID** of the data to an **ID** of the magnetic disk 8 as if the data were read from the magnetic disk 8, and sends a response to **Fibre Channel** 15.

When the address is not found in the memory 5 or the magnetic disk 6, the transmission **monitoring** and controlling apparatus 2 does not operate. In this case, the data are read from the magnetic disk 9 because the transmission **monitoring** and controlling apparatus 2 does not operate. The data which are read out from the magnetic disk 9 are outputted to **Fibre Channel** 15, and sent back to the first system 1.

For example, when the processing controller 22 obtains the data from the memory 5 or the magnetic disk 6, the processing controller 22 sends a response to the first system 1. At this time, the processing controller 22 changes the transmission origin port **ID** to a port **ID** of the magnetic disk 9. Since the first system 1 has received the response, even if the first system 1 receives a response from the magnetic disk 9, the first system 1 ignores a response from the magnetic disk 9.

As stated, even if the magnetic disk connected to **Fibre Channel** is in the maintenance operation, the first system is able to output a command to write without recognizing the maintenance operation. The transmission **monitoring** and controlling apparatus stores data for the magnetic disk which is in the

maintenance operation. After the maintenance operation, the transmission **monitoring** and controlling apparatus generates a command to write the stored data back on the magnetic disk which was in the maintenance operation, and outputs the command to the network.

Practically, the transmission **monitoring** and controlling apparatus 2 changes its port **ID** to the port **ID** of the magnetic disk 8. Hence, packets which are outputted by the transmission **monitoring** and controlling apparatus 2 have the port **ID** of the magnetic disk 8. Since the RAID level is assumed to be 5 in embodiment 5, redundant data for the data to be read from the magnetic disk 8 are stored in magnetic disk 7 or 9. Therefore, the processing controller 22 **monitors** the error report from the magnetic disk 8, and generates data which the first system 1 tries to read from the magnetic disk 8 by using data from magnetic disks 7 and 9. Then, the processing controller 22 outputs the generated data to **Fibre Channel** 15. At this time, the **ID** of the magnetic disk 8 is set as the transmission origin port **ID** of the packet. Hence, when the first system 1 receives the packet, the first system 1 is able to obtain the data as if the data were read from the magnetic disk 8.

Further, a spare disk, e.g., magnetic disk 11, which is connected to **Fibre Channel** 15 is able to be used for the magnetic disk 8. When lines are open in **Fibre Channel** 15, the transmission **monitoring** and controlling apparatus 2 generates a command to write the data of the magnetic disk 8 which are stored in the memory 5 on the magnetic disk 11. Then, the transmission **monitoring** and controlling apparatus 2 is able to operate the magnetic disk 11 for the magnetic disk 8. In that case, the transmission **monitoring** and controlling apparatus 2 changes the **ID** of the magnetic disk 11 to the **ID** of the magnetic disk 8, so that the magnetic disk 11 has the same address with the magnetic disk 8.

As stated, even if a failure has occurred in a magnetic disk connected to **Fibre Channel,** the transmission **monitoring** and controlling apparatus is able to regenerate data if the data are able to be generated from data in other magnetic disks. Further, the transmission **monitoring** and controlling apparatus is able to send a response to the first system as if the data are retrieved normally from the magnetic disk which has failed.

The above-stated network is an optical LAN (Local Area Network) or **Fibre Channel**. Therefore, the transmission **monitoring** and controlling apparatus of this invention is easily provided in an existing system which includes an optical LAN or **Fibre Channel** network.

The converter is able to convert ten bit information which is transmitted through the **Fibre Channel** to eight bit information. Therefore, the transmission **monitoring** and controlling apparatus is able to process eight bit information. The transmission **monitoring** and controlling apparatus is able to operate normally when it is connected to **Fibre Channel**. For example, since eight bit information is transmitted in a SCSI (Small Computer System Interface), the transmission **monitoring** and controlling apparatus is able to operate on the SCSI without converting the bit length of the data.

13. The transmission **monitoring** and controlling apparatus of claim 1, wherein the network is a **Fibre Channel**.

20. The transmission monitoring and controlling apparatus of claim 1 wherein the processing controller changes an origin port **ID** of the back-up copy retrieved from the controlling memory to an **ID** of the failed device.

21. The transmission monitoring and controlling method of claim 15 wherein transmitting the back-up copy retrieved from the controlling memory to the first system includes changing an origin port **ID** of the back-up copy to an **ID** of the failed device.